REMARKS

In response to the Office Action mailed on October 4, 2006, Applicants respectfully request reconsideration. In furtherance of prosecuting this patent application, Applicants submit the following remarks discussing patentability of rejected claims 1-43 as set out in the present Office Action and new claims 44-51.

Applicants encourage the Examiner to call the undersigned Attorney of record if the Examiner feels that a discussion of the pending claims would be helpful towards furthering prosecution of the present application.

Rejection of Pending Claims 1-43

The Examiner rejects claim 1 under 35 U.S.C. §103(a) as being unpatentable over Young (U.S. Patent 6,782,531) in view of Shenassa (U.S. Patent 6,842,856). The Office Action likens elements in Young and Shenassa to those in claim 1 to reject the claimed invention.

Applicants have reviewed the pending claims in view of the cited references and respectfully submit that the claimed invention includes distinguishing limitations over the cited art.

For example, the Examiner admits that Young does not disclose "obtaining identities of a plurality of plug-in modules; based on queries to the plurality of plug-in modules, retrieving a dependency list indicating respective plug-in services provided by, and required by, each plug-in module identified in the identities of a plurality of plug-in modules." The office action states that Shenassa discloses such limitations. Applicants respectfully disagree.

First, the cited passages in Shenassa (and as further shown in FIG. 5) merely indicate that a respective plug-in manager 120 can include a data structure for storing properties 191-193. The properties 191-193 include dependency information specifying an order in which the plug-ins should be started. The office action seems to be claiming that a plug-in manager is a plug-in module. Applicants respectfully submit that the cited passages in Shenassa provide no indication that the plug-in manager is a plug-in module that is queried to obtain dependency information indicating an ordering of plug-in module execution. Thus, the claimed invention includes limitations not found in the cited prior art.

Second, and perhaps more importantly, even if the plug-in manager was considered a plug-in module, Shenassa still would not disclose the limitations in claim 1. For example, claim 1 recites obtaining identities of a plurality of plug-in modules and thereafter querying each of the plug-in modules to learn of dependency information. At most, Shenassa only recites a single query to obtain the property information 191-193 to identify plug-in module order information. In other words, Shenassa includes a single data structure storing dependency information that is stored in the plug-in manager or is accessible by the plug-in manager. There would be no need to query a plurality of multiple plug-in modules as in the claimed invention because the property information 191-193 in Shenassa is stored in a single location accessible by a respective plug-in manager. Thus, the cited reference and claimed invention are not equivalent.

Moreover, the claimed invention recites retrieving dependency information from each of the queried plug-in modules. As discussed above, Shenassa only discloses obtaining dependency information from a plug-in manager (e.g., a single entity) that stores the data structure including properties 191-193, not multiple different plug-in modules as in the claimed invention. The claimed invention recites "retrieving a dependency list ... provided by ... each plug-in module" for the plurality of plug-in modules. Shenassa does not retrieve dependency information from each of a plurality of identified plug-in modules.

Note further that neither Shenassa nor Young recite calculating a plug-in initiation order based on a dependency list provided by each queried plug-in module as in the claimed invention. This aspect of the claimed invention enables an entity to calculate a plug-in order based on queries and a receipt of dependency information from multiple plug-in modules. Shenassa does not disclose calculating a plug-in order. Instead, Shenassa merely indicates that a single data structure storing properties 191-193 can be accessed to learn of dependency information.

This claimed technique enables the plug-in modules themselves to provide dependency information and what other plug-in modules need to be executed to carry out a respective function (e.g., service, application, etc.). Again, Applicants respectfully submit that merely accessing dependency information a single data structure (that does not reside in a plug-in module) as in Shenassa does not teach or suggest retrieving dependency information from a plurality of plug-in modules and calculating an initiation order as in the claimed invention. Thus, Applicants respectfully request the withdrawal of the respective rejection of claim 1 under 103(a).

For the reasons stated above, Applicants submit that claim 1 includes limitations not found in any of the cited references and therefore is patentably distinct and advantageous over the cited prior art. Applicants respectfully request the withdrawal of the rejection of claim 1 under 35 U.S.C. §103(a) or request that the Examiner more particularly point out passages in the prior art that teach the above limitations. Accordingly, Applicants respectfully request allowance of claim 1.

By virtue of dependency, Applicants respectfully submit that claims 2-14, 38-47 should be in condition for allowance as well.

Claim 15 includes similar limitations as discussed above for claim 1. Accordingly, Applicants respectfully submit that claim 15 includes similar patentable distinctions over the cited prior art as does claim 1. Applicants respectfully request allowance of claim 15 as well as corresponding dependent claims 16-28 and 48-51.

Claim 30 includes similar limitations as discussed above for claim 1. Applicants therefore respectfully request allowance of claim 30 over the cited prior art.

Applicants have amended claim 29 in accordance with the discussion above. Accordingly, claim 29 should be allowable for the same reasons as claim 1. By virtue of dependency, claims 31-37 also should be in condition for allowance.

Applicants respectfully submit that the pending dependent claims include further patentable distinctions over the cited art.

For example, claim 3 recites that retrieving the dependency list includes receiving a dependency response from each of multiple plug-in modules. Shenassa merely recites a technique of accessing a data structure (of dependency information) managed by a corresponding plug-in manager to identify an initiation order. Shenassa does not disclose communicating with corresponding plug-in modules specified in the data structure to learn of dependency information. Thus, Applicants respectfully request the withdrawal of the respective rejection of claim 3. Claim 17 includes similar limitations as claim 3 and should be allowable for similar reasons.

Claim 7 recites analyzing dependency information retrieved from multiple plug-in modules and creating a plug-in initiation order based on such information. Contrary to the Examiner's assertion, the cited prior art provides no such equivalent. Instead, the cited prior art teaches away from the claimed invention because it discloses accessing a single data structure including a pre-assembled listing of dependency information. There is no indication that the data structure is derived based on queries to each of multiple plug-in modules. Thus, Applicants respectfully request the withdrawal of the respective rejection of claim 7.

Claim 9 recites "forwarding, via a dependency available interface associated with a respective plug-in module, a list of initiated plug-in services of other plug-in modules that are currently available for use by the respective plug-in module." To reject this claim, the Examiner cites Young at column 8, lines 41-43 which states:

> Plug-ins 220 preferably have a system services interface for accessing system services provided via the framework 225, e.g., to allow the plug-ins 220 to read configuration data.

Applicants traverse the rejection on grounds that the cited passage in Young recites a respective technique that teaches away from the claimed invention.

The Examiner suggests that the claim does not indicate directionality. Applicants respectfully disagree. The "forwarding" is achieved "via a dependency available interface associated with a respective plug-in module." This means that the respective plug-in module forwards a list of initiated plug-in services currently available for use unlike the techniques in the cited prior art.

Claim 10 recites wherein "the step of initiating service operation of plug-in modules according to the plug-in initiation order comprises performing, for each respective plug-in module in the plug-in initiation order, the steps of:

determining, from a published list of services available by initiated plug-in modules, an identity of each initiated plug-in service required by the respective plug-in module;

forwarding to the respective plug-in module, via a dependency available interface associated with the respective plug-in module, the identity of each initiated plug-in service required by the respective plug-in module;

receiving a list of services initiated by the respective plug-in module; and

adding the list of services provided by the respective plug-in module to the published list of services."

The Office Action now cites Shenassa to reject this aspect of the claimed invention. Claim 10 recites receiving a list of services initiated by a respective plug-in module; and adding the list of services provided by the respective plug-in module to the published list of services. There is no indication in the cited passage that the application or any other resource receives information from a respective plug-in module, especially information such as services provided by the plug-in module for purposes of updating a respective information base.

This claimed technique (in claim 10) enables the plug-in modules to manage themselves based on knowing which plug-in module from which they depend. The plug-in modules in Shenassa are executed at different times based on accessing a pre-assembled execution order stored in a special data structure. The plug-in modules themselves do not know which other plug-in modules from which they depend. Thus, Applicants respectfully request that Examiner allow claim 10 over the cited prior art. Claim 24 includes similar limitations as claim 10 and should be allowable for similar reasons.

Claim 31 recites "querying a set of plug-in modules to identify services provided by the set of plug-in modules." The Examiner cites column 11, lines 60-65 in Asazu to reject the claimed invention, which reads as follows:

> In the preferred embodiment of the present invention, these services is separated from the package body as the QT plug-in component 14, and <u>the query interface package 12 itself is configured to provide only the management and/or look-up functions of the QT plug-in component 14</u>, permitting dynamic addition of various services in future.

Note that the query interface package 12 and plug-in component 14 are shown and discussed with respect to FIG. 3 of Asuza. The query interface package 12 and plug-in module 14 are separate entities. The query interface package 12 is an entity providing management and look-up with respect to the plug-in module 14. There is no indication that the query interface package 12 communicates with the plug-in module to identify services provided by the plug-in module 14.

The Examiner further cites the QT plug-in descriptor of FIG. 15 in Asazu. Applicants respectfully submit that there is no indication that such a resource (the QT plug-in descriptor) is provided by a respective plug-in module. Accordingly, Applicants respectfully submit that claim 31 includes limitations not taught or suggested by the cited prior art.

Claim 32 recites "in response to querying the set of plug-in modules, identifying plug-in modules not identified by the software application as being necessary but which are identified by the set of plug-in modules as being necessary to carry out execution of an operation on behalf of the software." The Examiner now cites Shenassa (column 7, lines 1-4) to reject this aspect of the claimed invention. Applicants respectfully assert that, contrary to the Examiner's assessment of the cited passage, there is no indication whatsoever of querying multiple plug-in modules. Further, there is no indication of carrying out a step of identifying plug-in modules not identified by the software application as being necessary but which are identified by the set of plug-in modules as being necessary to carry out execution of an operation on behalf of the software. In fact, as discussed above, the plug-in modules provide no such information in Shenassa. Accordingly, Applicants respectfully submit that claim 32 includes limitations not taught or suggested by the cited prior art.

New Claims 44-51

To expedite prosecution of the present application, Applicants submit new claims 44-51 that further emphasize the distinguishing features over the cited prior art.

Support for claims 44-51 can be found in FIG. 1 and corresponding text of the subject patent application. For example, plug-in manager process 150 generates queries 171 to plug-in modules 160 to retrieve dependency information 172. FIG. 2 and corresponding text describes this process of initiating queries and gathering responses from multiple plug-in modules. FIG. 5 and corresponding text illustrates the different plug-in modules 160 and corresponding dependency information. For example, plug-in module 160-1 (plug-in module A) provides service A and requires plug-in module E and B to operate; plug-in module B (160-2) provides service B and requires plug-in modules C and D; and so on. FIGS. 6 and 7 describe details of the retrieval of dependency information from each plug-in module and a technique of creating a dependency tree as shown and discussed in FIG. 8. Accordingly, a process can initially identify the need for plug-in module 401 and plug-in module 406. Based on querying each of these plug-in modules, the process can subsequently identify that these plug-in modules require plug-in modules 402, 403, and 407. Based on querying plug-in modules 402, 403, and 407, the process can subsequently identify that these second tier plug-in modules require plug-in modules 404, 405, and 408 as shown.

Applicants respectfully submit that none of the cited prior art discloses querying of multiple plug-in modules and utilizing the retrieved dependency information to produce a dependency tree. Instead, the (seemingly) most relevant reference (Shenassa) discloses access to a single, pre-assembled data structure identifying dependency information for a group of managed plug-in modules. However, as discussed above, this is not equivalent to or suggestive of the claimed invention.

CONCLUSION

In view of the foregoing remarks, Applicants submit that the pending claims as well as newly added claims are in condition for allowance. A Notice to this affect is respectfully requested. If the Examiner believes, after reviewing this Response, that the pending claims are not in condition for allowance, the Examiner is respectfully requested to call the Representative.

Applicants hereby petition for any extension of time which is required to maintain the pendency of this case. If there is a fee occasioned by this response, including an extension fee, that is not covered by an enclosed check, please charge any deficiency to Deposit Account No. 50-3735.

Respectfully submitted,

[illegible]

Paul P. Kriz, Esq.
Attorney for Applicant(s)
Registration No.: 45,752
Chapin Intellectual Property Law, LLC
Westborough Office Park
1700 West Park Drive

Westborough, Massachusetts 01581
Telephone: (508) 616-9660
Facsimile: (508) 616-9661

Attorney Docket No.: EMC01-11(01046)

Dated: January 4, 2007